

**REMARKS**

Claims 1-20 are pending in the application, with claims 1, 9, and 12 being the independent claims.

Applicant respectfully traverses the Examiner's rejection of each independent and dependent claim pending in the application.

***Rejections under 35 U.S.C. § 103***

Claims 1-20 are rejected under 35 U.S.C. 103(a) as being obvious in light of the prior art in the field.

Claims 1-20 are rejected as being unpatentable over U.S. Patent No. 6,298,434 (hereinafter "Lindwer") in view of U.S. Patent No. 6,606,743 (hereinafter "Raz"). Applicant respectfully traverses the rejection.

With respect to independent claims 1, 9 and 12, the combination of Lindwer and Raz does not teach or suggest the claimed invention. By Examiner's admission, Lindwer does not teach the technique of claims 1, 9 and 12 of determining an entry point into shared execution code based on the stack state.

Raz does not supplement Lindwer to teach or suggest the claimed invention. Raz does not teach the technique of claims 1, 9 and 12 of determining an entry point into shared execution code based on the stack state. Raz discloses using a memory mapped Intelligent Stack that is divided into four areas (Raz, col. 6 lines 54-67, col. 7 lines 1-67, col. 8 lines 1-67, col. 9 lines 1-7). The fourth area of this memory mapped Intelligent Stack contains registers that control the operation of the program language accelerator core. The program language accelerator core is a piece of hardware containing memory, a DMA controller, and a Arithmetic Logic Unit ("ALU"), which can be embodied on a single chip along with the CPU or on a separate chip (Raz, col. 12 lines 13-37). Java code is converted into native CPU instructions by the program language accelerator software, which translates certain arithmetic operations into memory write commands directed at specific addresses in the fourth area of the Intelligent Stack of the program language accelerator core (Raz, col. 12 lines 7-12). When executing the translated code the CPU will perform the memory write

commands, writing to the Intelligent Stack of the program language accelerator core. The program language accelerator core will then automatically perform the appropriate arithmetic operation based on the address written to in the Intelligent Stack using values from a different, specified part of the Intelligent Stack. For example, a command to write to address 16 of the Intelligent Stack will result in an addition operation being performed by the ALU of the language accelerator core.

Raz does not disclose sharing code between the CPU and the language accelerator core. Instead, the language accelerator core is responsive to instructions from the CPU, specifically to memory write commands issued by the CPU to certain specified memory address in the Intelligent Stack. Raz does not disclose determining an entry point into shared execution code, as there is no shared execution code. There is only one execution code, which is a translation from Java code. This execution code is translated by the language accelerator software and will therefore differ in length from the execution code produced for the same Java code by a different translator, but at no point is the code entered into anywhere but at the very beginning of the code. Raz employs a system of substituted code, substituting code performing write operations to certain memory addresses for code performing arithmetic operations. Write operations to these specified memory addresses trigger performance of certain arithmetic operations in specialized hardware. Further, the stack state is not used to determine an entry point into shared execution code. The stack state is used to trigger operations in hardware, specifically in the ALU attached to the program language accelerator core. The stack state does not determine an entry point into software code. This is wholly distinct from the method of shared execution code with an entry point determined by stack state employed by embodiments the present invention.

Therefore, Lindwer does not teach all of the limitations of the present application and the combination of Lindwer with Raz does not overcome the deficiencies of Lindwer. Independent claims 1, 9, and 12 are allowable over Lindwer in view Raz.

Claims 2-8, 10, 11, 13-20 depend from the independent claims which are allowable over Lindwer and Raz as discussed above.

### ***Conclusion***

All of the stated grounds of rejection have been properly traversed. Applicants therefore respectfully request that the Examiner reconsider all presently outstanding rejections and that they

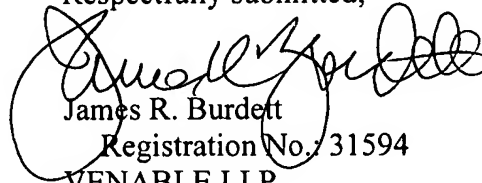
Application No. 10/813,599  
Amendment dated November 6, 2006  
Reply to Office Action of August 4, 2006

Docket No.: 42339-198432

be withdrawn. Applicants believe that a full and complete reply has been made to the outstanding Office Action and, as such, the present application is in condition for allowance. If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is hereby invited to telephone the undersigned at the number provided.

Dated: November 6, 2006

Respectfully submitted,

A handwritten signature in black ink, appearing to read "James R. Burdett", is written over the printed name and registration number.

James R. Burdett

Registration No. 31594

VENABLE LLP

P.O. Box 34385

Washington, DC 20043-9998

(202) 344-4000

(202) 344-8300 (Fax)

Attorney/Agent For Applicant